4
Metodologia da pesquisa - Efeito das abordagens

Neste capítulo se estudam as diferentes abordagens propostas por Newbold (1998) para a aplicação da metodologia da cadeia crítica e gestão de buffers de tempo – *CC/BM* – no ambiente de múltiplos projetos. A análise das características, vantagens e desvantagens de cada abordagem são realizadas através de um processo de comparação. A comparação é estruturada pelo "problema-exemplo" escolhido na literatura pesquisada e pelos diferentes "casos de estudo teóricos" criados a partir dele.

A finalidade de criar diferentes variações do mesmo problema é a geração de situações extremas que testem o desempenho dos principais elementos da metodologia dentro do cenário da abordagem utilizada. Essas variações são realizadas aproveitando a facilidade de análise do problema original e sem perder as características de interdependência de recursos que ele possui. A implementação dos problemas estudados é feita com a utilização da ferramenta Prochain - Pipeline da empresa Prochain Solutions Inc (*add-in* do Microsoft Project).

Essa análise preliminar permite, também, estabelecer uma base necessária para o entendimento das funções e efeitos que a implementação dos *buffers* de capacidade – BC tem sobre o sistema. Similarmente, este estudo fornece o alicerce para o entendimento das conseqüências de programação originadas pelas regras de priorização usadas na metodologia. O desenvolvimento desses estudos é feito no capítulo V.

As definições dos gráficos utilizados nesta seção são apresentadas no apêndice A.

4.1.
O problema-exemplo

4.1.1.
O Problema original

O problema-exemplo, proposto por Pritsker B. *et al.* (1969.), foi escolhido da literatura pesquisada por ser um sistema simples de múltiplos projetos. A estrutura da rede é constituída por três projetos com um pequeno número de atividades e relacionamentos do tipo *finish-start* (*i.e.* que a atividade só pode ser iniciada quando sua antecessora tenha terminado). Não obstante a simplicidade dos projetos, eles apresentam, de um ponto de vista global de múltiplos projetos, uma interdependência importante causada pelo compartilhamento de recursos limitados.

A Figura 4.1 mostra a programação ótima do sistema, a qual foi obtida mediante uma abordagem de programação inteira. Contra esse programa são contrastados os resultados do estudo. Contudo, é importante levar em consideração que o problema foi desenhado com base nos conceitos das metodologias PERT/CPM, pelo que se pressupõe que as durações de cada atividade levam uma margem de tempo de segurança embutida. Na aplicação da metodologia CC/BM se utilizaram as durações das atividades correspondentes à mediana (probabilidades de não exceder e de exceder iguais a 50%), ou seja, sem qualquer segurança embutida. Com essa lógica, e, para não diminuir as durações originais a ponto de perderem significância, se considera que o programa ótimo usado como referência comparativa corresponde ao dobro das durações originais, tal como apresentado na Figura 4.2

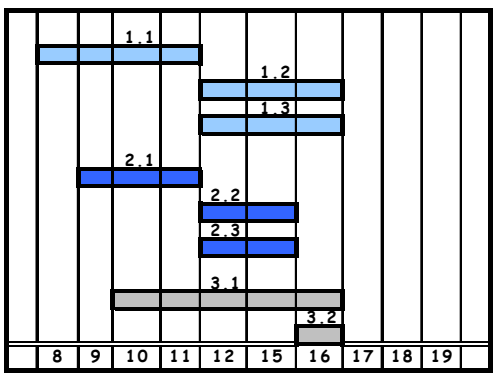

Figura 4.1: Programa ótimo original

Fonte: Pritsker B. *et al.* (1969.)

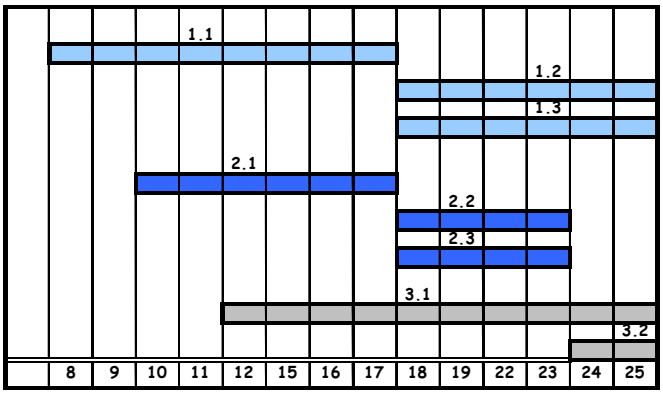

Figura 4.2: Programa ótimo modificado

Para a implementação do problema foi utilizada como data de início o dia 8/10, sendo que os projetos dois e três têm datas de chegada estabelecidas no dia 9 e 10 de outubro, respectivamente. A ferramenta de software utilizada na implementação respeita, sempre que sejam viáveis, as datas de início especificadas. Entretanto, as datas de entrega podem não ser respeitadas pela programação. Os dados da rede se encontram na Tabela 4.1.

Tabela 4.1: Dados das redes dos múltiplos projetos
Fonte: Fonte: Pritsker B. *et al.* (1969.)

Projeto	Atividade	Relação de Precedência	Chegada	Duração	Restrição de data de entrega	Recursos		
						A	B	C
1	1	Nenhuma	1 (8/10)	4	8 (17/10)	5	3	2
1	2	(1,1)	1 (8/10)	3	8 (17/10)	0	1	1
1	3	Nenhuma	1 (8/10)	3	8 (17/10)	2	0	2
2	1	Nenhuma	2 (9/10)	3	9 (18/10)	1	1	1
2	2	Nenhuma	2 (9/10)	2	9 (18/10)	2	0	0
2	3	(2,1)	2 (9/10)	2	9 (18/10)	2	2	0
3	1	Nenhuma	3 (10/10)	5	9 (18/10)	2	1	1
3	2	Nenhuma	3 (10/10)	1	9 (18/10)	1	3	0
Disponibilidade de Recursos						**8**	**5**	**4**

Para melhor entendimento do problema foram adicionadas duas atividades-fantasma (*dummy activities*) de duração zero, elas representam o início e fim de cada projeto. A representação das redes é definida na Figura 4.3

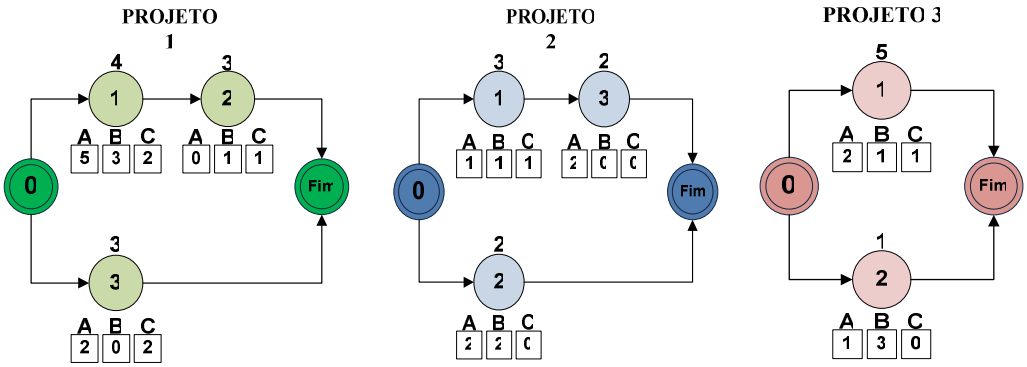

Figura 4.3: Representação gráfica das redes dos projetos.

As setas indicam os relacionamentos de precedência entre atividades, os números sobre os círculos são as durações das atividades e os números enquadrados correspondem às quantidades dos recursos que a atividade necessita para ser executada.

4.1.2. Casos de estudos teóricos: Variação do problema original

A variação do problema original foi desenvolvida com a finalidade de testar o comportamento das diferentes abordagens em situações prováveis de acontecer no mundo real. A análise é estruturada em dois casos de estudo, chamados de "teóricos" por serem desenvolvidos a partir de um problema criado para pesquisas teóricas.

Alem de analisar o comportamento das abordagens em condições normais (definidas na continuação), elas também são avaliadas em situações onde a definição das prioridades dos projetos é conflitante com as respectivas datas de chegada. A análise também é feita na etapa de execução, na qual se provocaram atrasos em algumas atividades da rede. Embora tais distúrbios não sejam exatamente iguais entre os casos, procuram ter o mesmo efeito e ser coerentes com a lógica do programa.

Entre os casos de estudo teóricos temos:

- **Caso 1. - Programação Normal**: Utiliza os parâmetros do *software* estabelecidos por padrão, considerando-se que desde o início da programação básica se conhece com certeza a chegada dos três projetos em diferentes datas, vale dizer, o programa contém os três

projetos todo o tempo. Respeitam-se as prioridades definidas originalmente, considerando que o projeto mais prioritário é o primeiro em carteira. O grau de importância diminui de acordo com a ordem de chegada.

- **Caso 2. - Programação Prioritária:** Difere do caso anterior no ordenamento prioritário dos projetos. Neste caso, a prioridade arbitrariamente estabelecida é preemptiva em relação à prioridade de ordem de chegada. Igualmente trabalha-se o tempo todo com os três projetos.

4.2. As abordagens de aplicação

Segundo Newbold (1998) a programação de múltiplos projetos por meio da CC/BM pode ser implementada seguindo três tipos de abordagens.

4.2.1. Abordagem de "Todos juntos"

Programa todas as atividades dos projetos em forma simultânea dentro de um único projeto criado artificialmente. Esse projeto artificial tem uma única data final que representa o término de todos os projetos. No programa artificial não se distingue explicitamente cada projeto, sendo que a programação das atividades é feita segundo o algoritmo da metodologia para projetos individuais. Isso implica a inserção de um ou mais *BA* e um único *BP*. Para facilitar a análise da programação decidiu-se não incluir *BR*, tornando o programa mais compacto e simples de analisar.

4.2.2. Abordagem de "Projetos sucessivos"

Essa abordagem parte dos projetos individuais corretamente programados segundo as recomendações da metodologia. Quer dizer que, supondo que existe uma disponibilidade total de recursos, em cada projeto se resolvem os conflitos de recursos, se define a cadeia crítica e se inserem os *buffers*.

O sistema que gerencia a programação desses projetos individuais é chamado de **Master**, este é definido como um gerenciador de projetos simultâneos que além de permitir a programação e o acompanhamento individual de cada projeto, equilibra a carga de trabalho do sistema e gera informações sobre o estado do recurso gargalo. A interação dessas funções faz com que o objetivo de manter equilibrado o fluxo de projetos, atingindo as datas prometidas seja mais factível.

A abordagem estabelece que o nivelamento de carga de trabalho do sistema é através de todos os recursos, ou seja, que não é permitido sobrecarga em qualquer recurso. Segundo Newbold (1998) à medida que o número de projetos simultâneos aumenta e o grau de complexidade, em função das dependências de recursos, se eleva, a eficiência do programa irá se deteriorando, podendo ser impossível atingir o nivelamento adequado de todos os recursos.

4.2.3.
Abordagem do "Recurso estratégico ou gargalo"

Essa abordagem tem a mesma concepção da aproximação anterior, também é desenvolvida no programa máster. Contudo, esta última se diferencia por resolver os conflitos de recursos entre projetos unicamente para o recurso definido como gargalo do sistema, permitindo-se a existência de sobrecargas. A abordagem supõe que ao resolver os conflitos do gargalo se criam espaços no programa que diminuem os conflitos dos outros recursos. Quanto a isso, Newbold (1998) adverte que são necessários procedimentos alternativos de gestão e programação que mitiguem o efeito desses conflitos.

4.3.
Implementação dos casos de estudo

O caso da programação normal é aplicado nas três abordagens. Já o caso de programação prioritária é aplicado unicamente na abordagem de "projetos sucessivos". A aplicação da programação prioritária tem características similares nas abordagens dos "projetos sucessivos" e do "recurso gargalo", razão pela qual só é desenvolvida na primeira delas. Pelas características de projeto individual que

a abordagem de "todos juntos" tem, não fazem sentido as variações do tipo de programação prioritária.

4.3.1. Abordagem de "Todos juntos"

4.3.1.1. Programação normal

Nessa abordagem, a implementação do programa base é relativamente simples e fácil de estruturar, não obstante, tal facilidade origina um efeito contrário na correta interpretação e consecução de objetivos individuais (de cada projeto) e globais (do sistema).

A estrutura do programa "fictício" contém todas as atividades dos três projetos, incluindo as "*dummy*" ou "*milestone*" de início e fim. A cadeia crítica, que não chega a ser representativa dos três projetos, é formada unicamente pelas duas primeiras atividades do projeto um e pelo *milestone* de término do projeto artificial. Isso faz com que se insiram um *buffer* de projeto- *BP* e cinco *buffers* de alimentação – *BA*. Estes, ao não ter espaço suficiente no programa, são inseridos com algum grau de consumação, que é compensado aumentando-se o tamanho do BP. Esse procedimento, aplicado no caso de projetos individuais, é eficiente para assegurar a proteção do projeto, evitando a origem de novos conflitos de recursos.

Contudo, sua aplicação em múltiplos projetos não é bem sucedida pela distorção das datas de término individuais, as quais são igualadas a uma mesma data. (25/10 – superior ao ótimo do projeto dois do artigo original, 23/10).

Usando a ferramenta *"toggle deadline"* (setas vermelhas), que permite visualizar a porção do BP que corresponderia a cada atividade, pode definir-se a segurança que as atividades dos projetos precisam. Isto é, pode-se definir a data de entrega para cada projeto.

O efeito negativo da não representatividade da estrutura da cadeia crítica se estende à etapa da execução, na qual se originam discrepâncias entre os relatórios de acompanhamento do projeto e o estado real do consumo do BP. Isso pode ser observado na Figura 4.4 onde o relatório da coluna 7 (*check list*) indica que as primeiras atividades de cada projeto têm maior prioridade de programação.

Contudo, ao não pertencerem à cadeia crítica, é muito provável que elas não sejam consideradas nas decisões de execução.

Essas discrepâncias se acentuam na presença de atrasos na execução do programa, no qual, dar prioridade de programação às atividades críticas (o que é correto segundo a metodologia), nem sempre assegura o bom desempenho do projeto. Pior ainda, pode ocasionar atrasos maiores dos projetos. Veja bem na Figura 4.5, como ao atrasar as primeiras atividades (não criticas) dos projetos dois e três ocasiona que o programa projetado para o dia 15/10 (*Monday*) priorize a programação da segunda atividade crítica do projeto um, fazendo com que as atividades do projeto dois sejam programadas três dias depois (18/10-*Thursday* e 19/10-*Friday*) e que o BP seja altamente consumindo.

Mesmo que essa operação gere um consumo do BP importante, ele não compromete a data final do programa fictício. Não obstante, o projeto dois se vê muito danificado em relação a seu programa base (determinado pelo termino marcado pelas setas vermelhas).

Contrariamente ao esperado, gerar esse mesmo atraso numa das atividades críticas (Figura 4.6), faz com que o dano no sistema e nos projetos seja muito menor.

Certamente, em outras condições o resultado seria diferente. Não obstante, a análise se interessa em demonstrar que a abordagem não considera a complexidade da interação dos três projetos. Assim ela gera um efeito negativo na estabilidade da rede e na qualidade de informação que ela oferece. As decisões de programação passam de incongruentes e confusas (nas etapas de planejamento e de execução em condições normais), a erráticas (quando o sistema é afetado por atividades atrasadas). Tal deficiência informativa do programa é conseqüência da pouca representatividade que os elementos chave da metodologia para projetos individuais (Cadeia crítica e *buffers* de tempo) têm para o controle de múltiplos de projetos, fazendo com que as vantagens da CC/BM sejam perdidas.

ID	Task Name	Focus Dur	% Impact	eductic Impact	Days Slack	Check Task	Chain Left	Orig. Chain
1	1 Cero	0 days	0	0	0			
2	**1 Atividade um**	**4 days**	**0**	**0**	**0**			
3	**1 Atividade dois**	**3 days**	**0**	**0**	**0**			
4	1 Atividade tres	3 days	0	0	0			
5	FB|1 Atividade tres-4|fim-16	2 days	100	0	0	7-7,2-2;2 Atividade 1	3d	3d
6	2 Cero	0 days	0	0	0			
7	2 Atividade 1	3 days	0	0	0			
8	2 Atividade 2	2 days	0	0	0			
9	FB|2 Atividade 2-8|fim-16	1 day	0	0	0	7-7,2-2;2 Atividade 1	2d	2d
10	2 Atividade 3	2 days	0	0	0			
11	FB|2 Atividade 3-9|fim-16	3 days	67	0	0	7-7,2-2;2 Atividade 1	5d	5d
12	3 Cero	0 days	0	100	0			
13	3 Atividade 1	5 days	0	100	0			
14	FB|3 Atividade 1-12|fim-16	3 days	100	0	0	12-11;3 Cero	5d	5d
15	3 Atividade 2	1 day	0	0	0			
16	FB|3 Atividade 2-13|fim-16	1 day	100	0	0	7-7,2-2;2 Atividade 1	1d	1d
17	**fim**	**0 days**	**0**	**100**	**0**			
18	PB|fim-16	7 days	0	0	0	12-11,7-7,2-2;2 Atividade 1	6d	7d

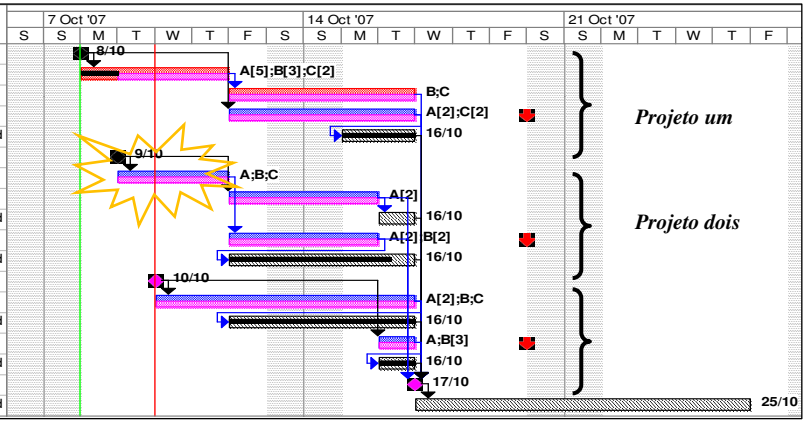

Figura 4.4: Incongruência entre importância de atividades e definição da cadeia crítica. Status ao 9/10

A atividade ressaltada é a atividade que segundo o relatorio *check task* é a atividade que poderia causar maior consumo do BP. A cadeia crítica está formada pelas duas primeiras atividades vermelhas. As setas vermelhas indicam a percentagem do BP que corresponde á atividade, no caso indicaria o que seria o término do projeto com a proteção embutida

ID	Task Name	Focus Dur	% Impact	eductic Impact	Days Slack	Check Task	Chain Left	Orig. Chain
1	1 Cero	0 days	0	0	0			
2	**1 Atividade um**	**5 days**	**57**	**0**	**0**			
3	**1 Atividade dois**	**3 days**	**57**	**0**	**0**			
4	1 Atividade tres	3 days	57	0	0			
5	FB|1 Atividade tres-4|fim-16	2 days	150	0	0	2-2;1 Atividade um	3d	3d
6	2 Cero	0 days	0	0	0			
7	2 Atividade 1	5 days	57	0	0			
8	2 Atividade 2	2 days	-14	0	5			
9	FB|2 Atividade 2-8|fim-16	1 day	0	0	0	8-8;2 Atividade 2	2d	2d
10	2 Atividade 3	2 days	57	0	0			
11	FB|2 Atividade 3-9|fim-16	3 days	233	0	0	2-2;1 Atividade um	3d	5d
12	3 Cero	0 days	0	100	0			
13	3 Atividade 1	6 days	29	100	2			
14	FB|3 Atividade 1-12|fim-16	3 days	167	0	0	2-2;1 Atividade um	4d	5d
15	3 Atividade 2	1 day	-14	0	5			
16	FB|3 Atividade 2-13|fim-16	1 day	0	0	0	2-2;1 Atividade um	1d	1d
17	**fim**	**0 days**	**57**	**100**	**0**			
18	PB|fim-16	7 days	57	0	0	2-2;1 Atividade um	7d	7d

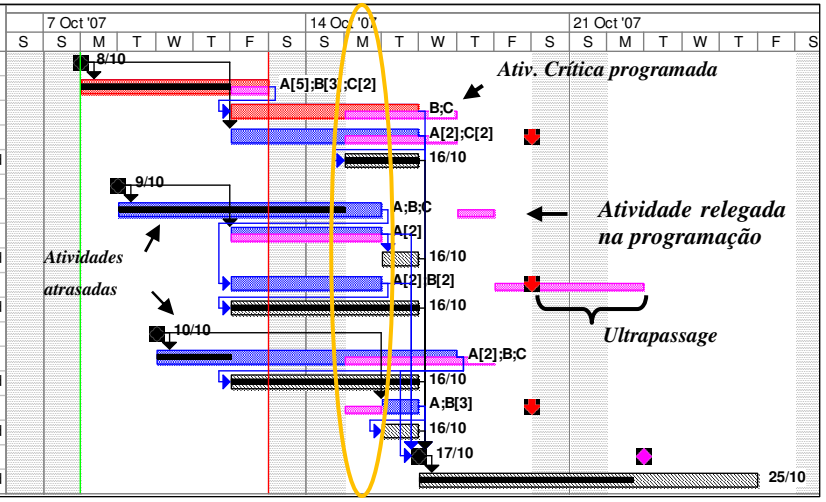

Figura 4.5: Atrasos nos projetos dois e três. Status no dia 12/10

A atividade 1 do projeto 2 (primeira marcada) é atrasada em dois dias. A atividade 1 do projeto 3 é atrasada em um dia (segunda marcada). Repare-se no dia 15/10 (marcado em laranja) como o programa projetado (linhas roxas), ao dar preferência à segunda atividade crítica, relega a atividade 1 do projeto 2 gerando o consumo do BP e a ultrapassagem do que seria sua data de término programada (seta vermelha).

ID	Task Name	Focus Dur	% Impact	eductic Impact	Days Slack	Check Task	Chain Left	Orig. Chain
1	1 Cero	0 days	0	0	0			
2	**1 Atividade um**	**6 days**	**29**	**0**	**0**			
3	**1 Atividade dois**	**3 days**	**29**	**0**	**0**			
4	1 Atividade tres	3 days	29	0	0			
5	FB|1 Atividade tres-4|fim-16	2 days	200	0	0	8-8,2-2,7-7;2 Atividade 2	3d	3d
6	2 Cero	0 days	0	0	0			
7	2 Atividade 1	5 days	29	0	0			
8	2 Atividade 2	2 days	29	0	0			
9	FB|2 Atividade 2-8|fim-16	1 day	0	0	0	8-8;2 Atividade 2	2d	2d
10	2 Atividade 3	2 days	29	0	0			
11	FB|2 Atividade 3-9|fim-16	3 days	133	0	0	7-7,8-8,2-2;2 Atividade 2	3d	5d
12	3 Cero	0 days	0	100	0			
13	3 Atividade 1	5 days	29	100	0			
14	FB|3 Atividade 1-12|fim-16	3 days	167	0	0	8-8,2-2,7-7;2 Atividade 2	3d	5d
15	3 Atividade 2	1 day	29	0	0			
16	FB|3 Atividade 2-13|fim-16	1 day	300	0	0	7-7,8-8,2-2;2 Atividade 2	1d	1d
17	**fim**	**0 days**	**29**	**100**	**0**			
18	PB|fim-16	7 days	29	0	0	8-8,2-2,7-7;2 Atividade 2	5d	7d

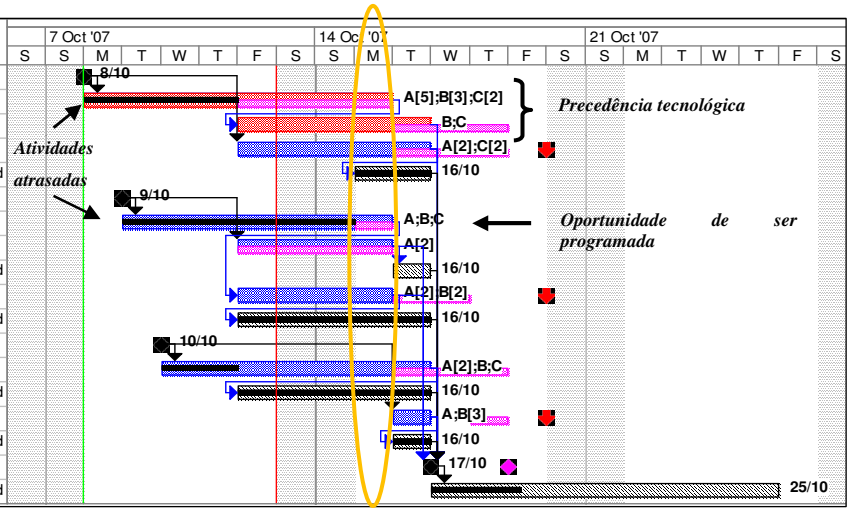

Figura 4.6:Atrasos nos projetos um e dois. Status ao dia 12/10

A atividade 1 do projeto 1 é atrasada um dia. A atividade 1 do projeto 2 é atrasada dois dias. Repare que o atraso no projeto 1 faz com que pelas dependências de precedência, a atividade 2 do projeto 1 (atividade crítica) não possa ser programada no dia 15/10, dando oportunidade de programação à atividade 1 do projeto 2. Isso faz que o BP seja menos consumido e as datas de entrega de projetos (setas vermelhas) não sejam ultrapassadas.

.

4.3.2.
Abordagem de "Projetos sucessivos"

4.3.2.1.
Programação normal

Essa abordagem estabelece um método de programação de múltiplos projetos totalmente diferente da abordagem anterior. A diferença tem origem na utilização do máster como cenário para o desenvolvimento da metodologia CC/BM. Certamente, o fato de que essa abordagem parte de programas individuais corretamente programados, permitindo o nivelamento da carga de trabalho entre eles sem alterar sua solução interna, faz com que alguns aspectos negativos da abordagem anterior sejam eliminados e se atinja um programa mais adequado para a administração de múltiplos projetos simultâneos.

A Figura 4.7 mostra como o máster, ao permitir a presença ativa e explicita das cadeias críticas, os BA, e, os BP dos três projetos, faz com que o sistema considere os objetivos individuais. Entretanto, para a consecução do objetivo global, (estabilidade do sistema) o máster exige que os começos dos projetos dois e três sejam postergados. Este deslocamento mitiga o efeito das interdependências entre os projetos, aumentando o *makespan* ou duração total do programa. Isso poderia ser visto como uma incongruência do programa com os objetivos teóricos da metodologia; não obstante, é importante esclarecer que a preocupação principal da metodologia é definir datas de entrega que sejam realmente factíveis. O objetivo de atingir um programa de múltiplos projetos com o mínimo *makespan* é subordinado à satisfação desse objetivo principal. Repare-se que a data final do conjunto de projetos não é superior ao programa ótimo original. (Figura 4.2)

Contudo, os efeitos originados pelas interdependências constituem um problema ainda mal solucionado. Entende-se por interdependências aos efeitos ocasionados num projeto por distúrbios (atrasos, re-processos, etc.) ocorridos num outro projeto. A origem desses efeitos está no compartilhamento de recursos entre projetos.

Para evitar todo tipo de sobrecargas a ação gerencial é mantida também durante a etapa da execução. Nela o programa projetado é atualizado com base no nivelamento completo (em todos os recursos). Com esta lógica se definem duas formas de realizar o acompanhamento e controle da execução dos projetos: sem alterar o programa de base, mantendo o tamanho original dos *buffers* (obtidos na programação individual); ou reprogramando algum ou todos os projetos do master cada vez que ocorre uma mudança significativa em relação ao programado no sistema. Segundo Herroelen & Leus (2001) é recomendado fazer reprogramações apenas se o BP estiver fora de controle. Já a teoria da CC/BM considera que o programa original foi elaborado para suportar qualquer distúrbio (não excepcional), sendo desnecessário reprogramá-lo (LEACH, 2005).

Certamente, para controlar os efeitos das interdependências, é necessário manter o programa original. A reprogramação do sistema implicaria zerar os *buffers* e recalcular a rede, perdendo a informação do estado dos projetos em relação ao plano original. A informação do consumo dos *buffers* é o único meio que permite inferir que atividade fora do projeto poderia ser a causa de atrasos. Voltando ao exemplo, a Figura 4.7 mostra o programa master com os recursos nivelados. Repare-se como a mesma atualização aplicada na Figura 4.8 origina informações importantes relativas às interdependências na Figura 4.9, na qual pode se observar como o projeto um afeta o consumo dos BP dos projetos dois e três, os que sofrem um consumo de 67% antes mesmo de começar a executar os projetos.

Ao ter os três projetos em evidência, reparou-se que o programa projetado é atualizado segundo as prioridades de projetos definidas na etapa de programação, sem levar em consideração o estado de consumo dos BP, contrariamente às considerações feitas por Cohen *et al.* (2004). No experimento que esses autores fizeram para comparar o desempenho da CC/BM, fazem que a escolha das atividades a executar foi baseada no estado dos *BP*, quer dizer, que as prioridades de programação mudavam dinamicamente com a execução dos projetos. Essa lógica é aplicada na transição do programa da Figura 4.10 ao programa da Figura 4.11, onde ignorando o programa projetado sugerido, executam-se prioritariamente as atividades dos projetos com BP mais consumidos. Essa

decisão não só faz com que os projetos dois e três cumpram com as datas de entrega prometidas, mas também diminui a duração efetiva do projeto. Isso, claro, sem considerar que novos atrasos possam acontecer no sistema. Este resultado deixa a dúvida sobre as razões pelas quais a ferramenta de *software* mantém fixa as prioridades das atividades e se, efetivamente, atualizá-las dinamicamente, tornaria o programa mais eficiente. O esclarecimento destas questões é abordado no Capítulo V.

ID	Task Name	Focus Dur	% Impact	Reductior Impact	Days Slack	Check Task	Chain Left
1	**Multiproject_Pristsker**	**11 days**	**0**	**0**	**0**		
1	Cero	0 days	0	0	0		
2	**Atividade um**	**4 days**	**0**	**100**	**0**		
3	**Atividade dois**	**3 days**	**0**	**100**	**0**		
4	Atividade tres	3 days	-100	0	4		
5	FB|Atividade tres-3|Fim-5	2 days	-100	0	0	4-3;Atividade tres	3d
6	**Fim**	**0 days**	**0**	**100**	**0**		
7	PB|Fim-5	4 days	0	0	0	2-1;Atividade um	7d
2	**Multiproject_Pristsker 2**	**8 days**	**0**	**0**	**0**		
1	**Cero**	**0 days**	**0**	**100**	**0**		
2	**Atividade 1**	**3 days**	**0**	**100**	**0**		
3	Atividade 2	2 days	-100	0	3		
4	FB|Atividade 2-3|Fim-5	1 day	-200	0	0	1-1;Cero	2d
5	**Atividade 3**	**2 days**	**0**	**100**	**0**		
6	**Fim**	**0 days**	**0**	**100**	**0**		
7	PB|Fim-5	3 days	0	0	0	1-1;Cero	5d
3	**Multiproject_Pristsker 3**	**8 days?**	**0**	**0**	**0**		
1	**Cero**	**0 days**	**0**	**0**	**0**		
2	**Atividade 1**	**5 days**	**0**	**0**	**0**		
3	Atividade 2	1 day?	0	100	0		
4	FB|Atividade 2-3|Fim-4	1 day	100	0	0	3-3;Atividade 2	1d
5	**Fim**	**0 days**	**0**	**100**	**0**		
6	PB|Fim-4	3 days	0	0	0	3-3,1-1;Cero	5d

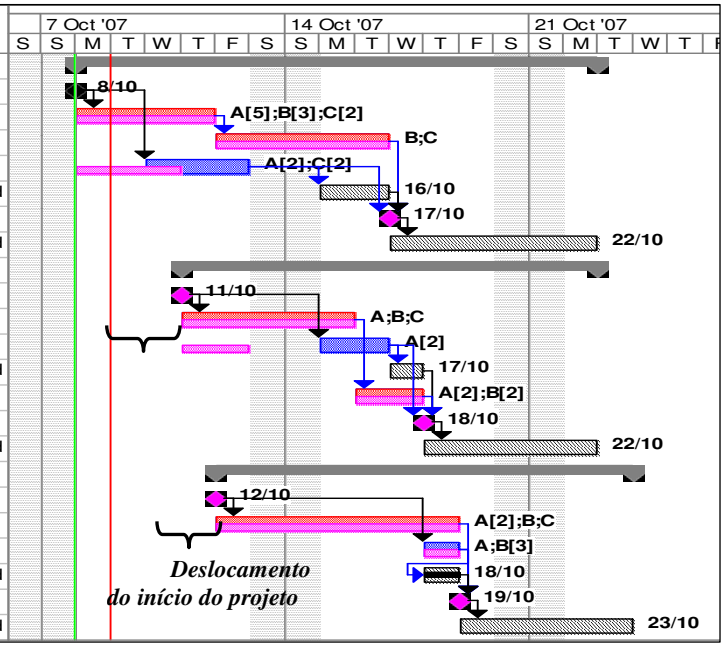

Figura 4.7 : Programa máster nivelando todos os recursos. Status do dia 09/10

No máster cada projeto esta diferenciado pelas linhas cinza escuras que indicam o início de um novo projeto. Cada projeto contém: cadeia crítica (retângulos vermelhos), BA e BP (retângulos hachurados). O máster também indica o *programa de linha de base* (tudo o que não é da cor roxa) e o *programa projetado* (linhas roxas). Os projetos dois e três são deslocados no tempo devido ao nivelamento de todos os recursos definindo um *makespan* de 12 dias (do 8/10 ao 23/10 sem considerar finais de semana).

ID	Task Name	Focus Dur	% Impact	Reduction Impact	Days Slack	Check Task	Chain Left	Orig. Chain
1	**Multiproject_Pristsker**	**11 days**	**0**	**0**	**0**			
1	Cero	0 days	0	0	0			
2	**Atividade um**	**6 days**	**49**	**100**	**0**			
3	**Atividade dois**	**3 days**	**49**	**100**	**0**			
4	Atividade tres	5 days	-25	0	2			
5	FB|Atividade tres-3|Fim-5	2 days	50	0	0	4-3;Atividade tres	2d	3d
6	**Fim**	**0 days**	**49**	**100**	**0**			
7	PB|Fim-5	4 days	49	0	0	2-1;Atividade um	4,98d	7d
2	**Multiproject_Pristsker 2**	**#######**	**0**	**0**	**0**			
1	Cero	0 days	0	0	0			
2	**Atividade 1**	**3 days**	**0**	**100**	**0**			
3	Atividade 2	2 days	-100	0	3			
4	FB|Atividade 2-3|Fim-5	1 day	-200	0	0	3-3;Atividade 2	2d	2d
5	**Atividade 3**	**2 days**	**0**	**100**	**0**			
6	**Fim**	**0 days**	**0**	**100**	**0**			
7	PB|Fim-5	3 days	0	0	0	2-2;Atividade 1	5d	5d
3	**Multiproject_Pristsker 3**	**#######**	**0**	**0**	**0**			
1	Cero	0 days	0	0	0			
2	**Atividade 1**	**5 days**	**0**	**0**	**0**			
3	Atividade 2	1 day?	0	100	0			
4	FB|Atividade 2-3|Fim-4	1 day	100	0	0	3-3;Atividade 2	1d	1d
5	**Fim**	**0 days**	**0**	**100**	**0**			
6	PB|Fim-4	3 days	0	0	0	3-3,2-2;Atividade 1	5d	5d

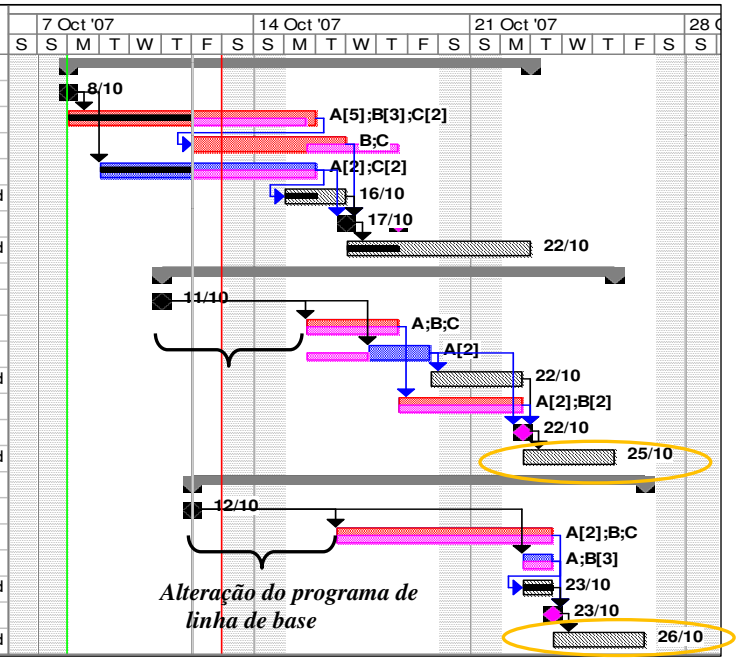

Figura 4.8: Reprogramando o máster. Status do 12/10

As atividades 1 e 3 do projeto 1 são atrasadas em um dia. Ao reprogramar o sistema o *programa de linha de base* é alterado, alterando tanto datas de término (um dia mais tarde - 26/10) como o estado dos BP dos projetos 2 e 3.O programa não da indícios do atraso nem de seu efeito no sistema.

Task Name	Focus Dur	% Impact	Reductior Impact	Days Slack	Check Task	Chain Left	Orig. Chain
Multiproject_Pristsker	11 days	0	0	0			
Cero	0 days	0	0	0			
Atividade um	6 days	49	100	0			
Atividade dois	3 days	49	100	0			
Atividade tres	5 days	-50	0	3			
FB|Atividade tres-3|Fim-	2 days	0	0	0	4-3;Atividade tres	1d	3d
Fim	0 days	49	100	0			
PB|Fim-5	4 days	49	0	0	2-1;Atividade um	4.98d	7d
Multiproject_Pristsker 2	8 days	0	0	0			
Cero	0 days	0	0	0			
Atividade 1	3 days	67	100	0			
Atividade 2	2 days	-33	0	3			
FB|Atividade 2-3|Fim-5	1 day	0	0	0	3-3;Atividade 2	2d	2d
Atividade 3	2 days	67	100	0			
Fim	0 days	67	100	0			
PB|Fim-5	3 days	67	0	0	2-2;Atividade 1	5d	5d
Multiproject_Pristsker 3	8 days?	0	0	0			
Cero	0 days	0	0	0			
Atividade 1	5 days	66	0	0			
Atividade 2	1 day?	67	100	0			
FB|Atividade 2-3|Fim -4	1 day	300	0	0	3-3;Atividade 2	1d	1d
Fim	0 days	67	100	0			
PB|Fim -4	3 days	67	0	0	3-3;Atividade 2	5.02d	5d

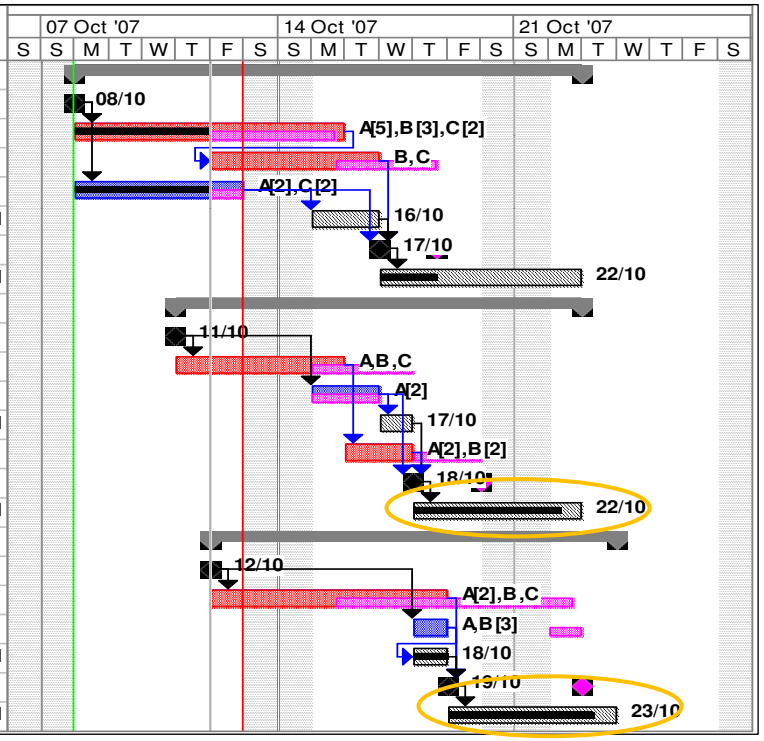

Figura 4.9: Sem reprogramar o máster. Status do 12/10

As atividades 1 e 3 do projeto 1 são atrasadas em um dia. O sistema mantém fixo o *programa de linha de base*, podendo-se observar o efeito do atraso no consumo dos BP dos projetos 2 e 3. Este consumo é gerado pela comparação do *programa projetado,* o qual programa as atividades o mais cedo possível, e o programa estabelecido na etapa de planejamento. Segundo o reporte dos BP o consumo registrado é de 67%, atingindo á área crítica de controle (> 50% de consumo).

Task Name	Focus Dur	% Impact	Reduction Impact	Days Slack	Check Task	Chain Left	Orig. Chain
Multiproject_Pristsker	11 days	0	0	0			
Cero	0 days	0	0	0			
Atividade um	6 days	51	100	0			
Atividade dois	3 days	51	100	0			
Atividade tres	5 days	-25	0	3			
FB|Atividade tres-3|Fim-	2 days	50	0	0	4-3;Atividade tres	1d	3d
Fim	0 days	51	100	0			
PB|Fim-5	4 days	51	0	0	2-1;Atividade um	4.02d	7d
Multiproject_Pristsker 2	8 days	0	0	0			
Cero	0 days	0	0	0			
Atividade 1	3 days	67	100	0			
Atividade 2	2 days	0	0	2			
FB|Atividade 2-3|Fim-5	1 day	100	0	0	3-3;Atividade 2	2d	2d
Atividade 3	2 days	67	100	0			
Fim	0 days	67	100	0			
PB|Fim-5	3 days	67	0	0	2-2;Atividade 1	4.01d	5d
Multiproject_Pristsker 3	8 days?	0	0	0			
Cero	0 days	0	0	0			
Atividade 1	5 days	34	0	0			
Atividade 2	1 day?	67	100	0			
FB|Atividade 2-3|Fim -4	1 day	301	0	0	3-3;Atividade 2	1d	1d
Fim	0 days	67	100	0			
PB|Fim -4	3 days	67	0	0	3-3;Atividade 2	4.99d	5d

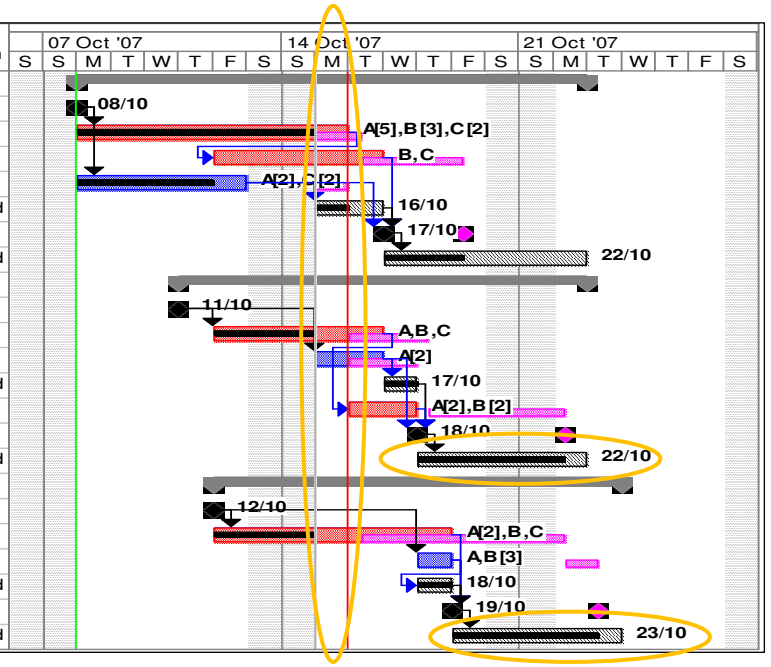

Figura 4.10: Programa projetado sugerido em base a prioridades fixas. Status ao dia 15/10

O programa projetado para o dia 15/10 sugere executar as atividades 1 e 3 do projeto 1. Os BP mostram o consumo que essa execução faria no sistema. Repare-se que esta programação é feita baseada nas prioridades definidas na etapa de planejamento (ordem de projetos 1,2 e 3) e que a disponibilidade de recursos não permite programar mais atividades. (Disp: A=8, B=5,C=4)

Task Name	Focus Dur	% Impact	Reduction Impact	Days Slack	Check Task	Chain Left
Multiproject_Pristsker	11 days	0	0	0		
Cero	0 days	0	0	0		
Atividade um	6 days	0	100	0		
Atividade dois	3 days	50	100	0		
Atividade tres	5 days	0	0	2		
FB|Atividade tres-3|Fim-	2 days	100	0	0	4-3;Atividade tres	1d
Fim	0 days	50	100	0		
PB|Fim-5	4 days	50	0	0	3-2;Atividade dois	3d
Multiproject_Pristsker 2	8 days	0	0	0		
Cero	0 days	0	0	0		
Atividade 1	3 days	33	100	0		
Atividade 2	2 days	0	0	0		
FB|Atividade 2-3|Fim-5	1 day	100	0	0	3-3;Atividade 2	2d
Atividade 3	2 days	33	100	0		
Fim	0 days	33	100	0		
PB|Fim-5	3 days	33	0	0	2-2;Atividade 1	2.99d
Multiproject_Pristsker 3	8 days?	0	0	0		
Cero	0 days	0	0	0		
Atividade 1	5 days	33	0	0		
Atividade 2	1 day?	33	100	0		
FB|Atividade 2-3|Fim -4	1 day	199	0	0	3-3;Atividade 2	1d
Fim	0 days	33	100	0		
PB|Fim -4	3 days	33	0	0	3-3,2-2;Atividade 1	3d

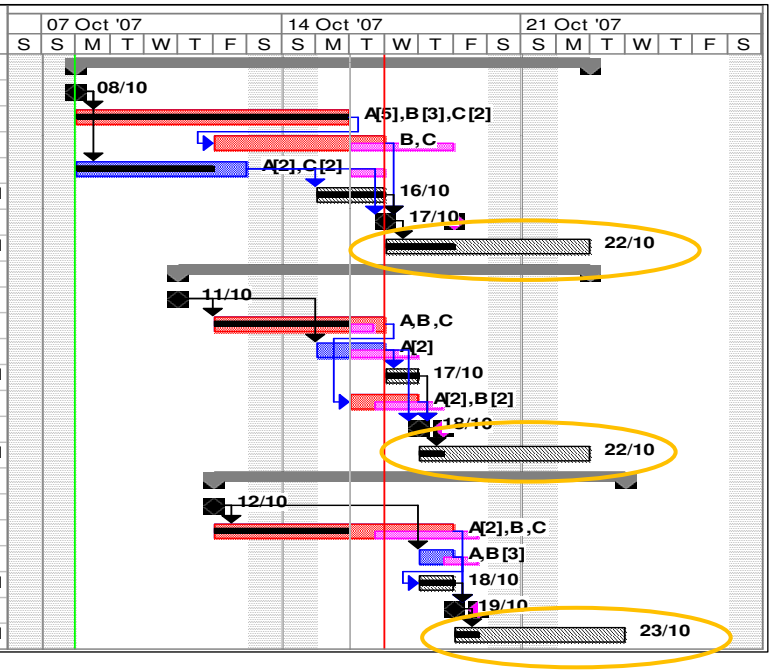

Figura 4.11: Programa resultante da aplicação de prioridades dinâmicas. Status do dia 16/10

O programa projetado para o dia 15/10 da figura anterior foi obviado, levando em consideração o estado dos BP se executaram as atividades que causavam o maior consumo. Isso é: atividade 1 do projeto 2, atividade 1 do projeto 3 e atividade 1 do projeto 1. Repare-se que se deu prioridade às atividades críticas dos projetos com maior consumo de BP, relegando a atividade 3 do projeto 1 por não ser crítica e estar protegida com um BA. O resultado da execução mostra um melhor resultado do que o previsto com a aplicação das prioridades fixas.

4.3.2.2. Programa Prioritário

Esse caso considera que o projeto três, mesmo tendo uma data de início posterior a do projeto dois, possui uma prioridade maior na programação do máster.

Ao inverter as prioridades entre os projetos dois e três se espera que os programas de ambos os projetos, em relação às datas de início, também sejam invertidas. Comparando o programa deste caso (Figura 4.12) com o resultante na programação normal *(*Figura 4.9) corroboramos que o projeto três é iniciado na mesma data do que o projeto dois, quando este tinha maior prioridade. Porém, projeto dois é programado um dia depois (15/10) da data de início do projeto três, quando este ocupava a menor prioridade.

O deslocamento do início do projeto dois é devido à programação antecipada da segunda atividade do projeto três, a qual passa de não ter folga (na programação normal) para ter três dias de folga. Isso faz com que a capacidade disponível dos recursos seja ocupada antecipadamente, provocando o atraso no projeto dois.

Esta programação é efeito da forma como se consideram as prioridades das atividades na etapa do nivelamento do master, no qual se assume que as atividades (críticas e não críticas) têm a mesma prioridade definida para o projeto ao qual pertencem. Desta maneira, acaba dando-se preferência na alocação de recursos às atividades não criticas que pertencem a projetos de maior prioridade.

Esse critério de priorização não considera que as atividades não críticas, pela natureza da metodologia, levam consigo um *buffer* de alimentação*,* ou seja, já estão protegidas para serem relegadas na programação. Ao programar essas atividades antes das atividades críticas geram-se folgas ociosas que implicam o atraso do início dos projetos seguintes.

Note-se na Figura 4.12 como o fato de programar mais tarde a atividade dois do projeto três não afetaria em nada à data de término do projeto, isso devido ao *BA* que protege à cadeia crítica e a data de entrega. Entretanto, tal atraso sim afetaria positivamente o término do projeto dois, devido à antecipação da data de início.

Esta análise identifica o critério utilizado na priorização das atividades que determinam o programa de base do sistema. Tal critério determina a programação considerando unicamente as prioridades relativas definidas para cada projeto, isto é, eliminando o efeito que as cadeias críticas têm sobre o programa. As conseqüências dessa lógica de programação são discutidas no Capítulo V

ID	Task Name	Focus Dur	% Impact	Reduction Impact	Days Slack	Check Task	Chain Left	Orig. Chain
1	**Multiproject_Pristsker**	**11 days**	**0**	**0**	**0**			
1	Cero	0 days	0	0	0			
2	**Atividade um**	**4 days**	**0**	**100**	**0**			
3	**Atividade dois**	**3 days**	**0**	**100**	**0**			
4	Atividade tres	3 days	-100	0	4			
5	FB|Atividade tres-3|Fim-5	2 days	-100	0	0	4-3;Atividade tres	3d	3d
6	**Fim**	**0 days**	**0**	**100**	**0**			
7	PB|Fim-5	4 days	0	0	0	2-1;Atividade um	7d	7d
2	**Multiproject_Pristsker 3**	**8 days?**	**0**	**0**	**0**			
1	**Cero**	**0 days**	**-33**	**0**	**1**			
2	**Atividade 1**	**5 days**	**0**	**100**	**0**			
3	Atividade 2	1 day?	-100	0	3			
4	FB|Atividade 2-3|Fim-4	1 day	-200	0	0	3-3;Atividade 2	1d	1d
5	**Fim**	**0 days**	**0**	**100**	**0**			
6	PB|Fim-4	3 days	0	0	0	2-2;Atividade 1	5d	5d
3	**Multiproject_Pristsker 2**	**8 days**	**0**	**0**	**0**			
1	**Cero**	**0 days**	**-133**	**0**	**4**			
2	**Atividade 1**	**3 days**	**0**	**100**	**0**			
3	Atividade 2	2 days	-133	0	4			
4	FB|Atividade 2-3|Fim-5	1 day	-200	0	0	3-3;Atividade 2	2d	2d
5	**Atividade 3**	**2 days**	**0**	**100**	**0**			
6	**Fim**	**0 days**	**0**	**100**	**0**			
7	PB|Fim-5	3 days	0	0	0	2-2;Atividade 1	5d	5d

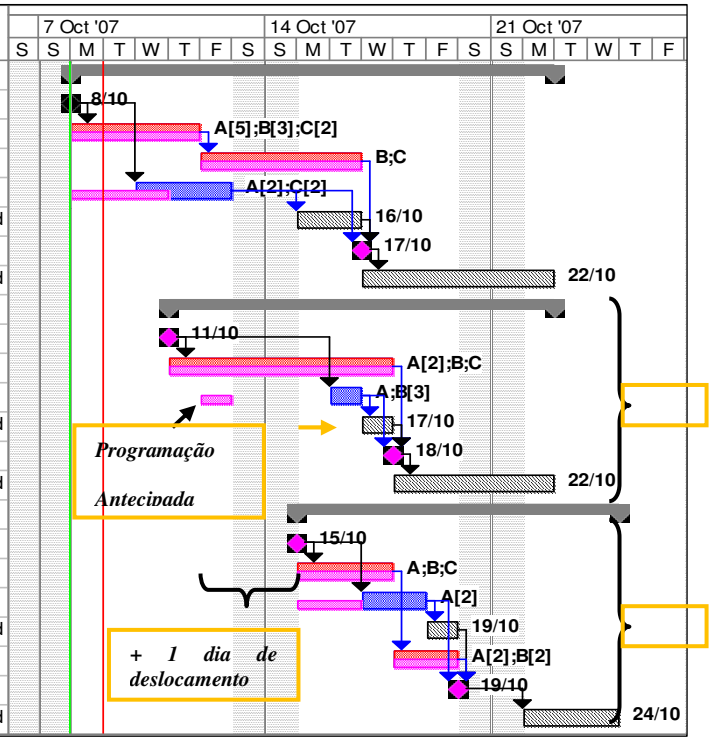

Figura 4.12: Programa máster segundo prioridades 1,3,2. Status ao dia 9/10

O projeto 2 é deslocado um dia mais da posição que ocupava o projeto 3 quando tinha a terceira prioridade. Isso é pela programação antecipada do ativ.2 do proj. 3 (que agora tem a prioridade dois). Repare-se que o programa deixa a ativ.2 do proj 3 com uma folga de 3 dias (ressaltada em laranja) ademais do espaço que o BA (um dia) gera. A antecipação da programação (no programa projetado) dessa atividade ocasiona que não exista mais disponibilidade do recurso B para programar o projeto dois

Atrasar o programa projetado do projeto 3 não afeta suas datas de termino, mas sim permite que a data de termino do projeto 2 seja menor num dia.

4.3.3.
Abordagem do "Recurso estratégico ou gargalo"

4.3.3.1.
Programa Normal.

Esse programa também é desenvolvido no cenário proporcionado pelo máster. A abordagem aplica fielmente as recomendações da teoria de restrições, definindo como única restrição do sistema, num dado momento, o recurso mais escasso. Contudo, a ferramenta de *software* utilizada permite definir mais de um recurso gargalo, tolerando a implementação de programas híbridos que interpolam entre "o programa do recurso estratégico" e "o programa dos projetos sucessivos", tal e como menciona Newbold (1998).

Para o exemplo, se define como gargalo do sistema, o recurso "A", por ser ele o recurso que inicialmente possui a maior carga de trabalho; além de ser o que tem participação no maior número de atividades. A Figura 4.13 mostra o máster resultante da abordagem. Repare-se que em comparação com a abordagem anterior, o projeto dois tem uma data de início mais cedo, implicando em recursos não gargalo sobrecarregados.

Nessa abordagem a programação se interessa por deixar ao recurso "A" livre de sobrecargas e folgas, é dizer, programam-se o maior número de atividades que o gargalo possa suportar (Figura 4.14). Neste contexto, os recursos "B" e "C" são deliberadamente sobrecarregados (Figura 4.15 e Figura 4.16) seguindo a lógica de que é mais conveniente, em simplicidade e economia, aumentar estes recursos do que deixar o gargalo folgado ou sobrecarregado. Para mitigar tais conflitos, a metodologia utiliza os *buffers* de capacidade – *BC*, que, embora tenham como função principal a proteção do recurso gargalo, também reduzem as sobrecargas dos recursos em conseqüência do espaço que o BC gera entre os projetos.

O programa da Figura 4.17 mostra o máster que considera BC's dimensionados a 25% (ver lógica de cálculo no apêndice A). Os efeitos que tais *buffers* têm sobre os recursos podem ser observados nos relatórios de carga de trabalho na Figura 4.18 e na Figura 4.19. Repare-se como a inserção do BC provoca que o excesso de carga no recurso "B" seja relegado no final do projeto e que a carga do recurso "C" seja totalmente eliminada. Repare-se também, que

essa solução faz com que as datas de entrega desses projetos sejam postergadas no tempo. Da mesma maneira, a data final do máster é postergada para o dia 26/10, ultrapassando o programa ótimo (23/10) apresentado por Pritsker B. *et al.* (1969.).

O dimensionamento do BC gera uma curva de *"trade-off"* na decisão de ter um sistema bem equilibrado e com altos níveis de proteção, ou um sistema de *makespan* curto, projetos menos deslocados no tempo, mas, com maior probabilidade de interferência entre eles. . Esse efeito explica que projetos mais isolados, com maiores BC, geram sistemas mais seguros e de maior duração. É possível que tais projetos sejam considerados como menos competitivos em situações de concorrência no mercado.

Certamente a localização ótima dentro desta curva de *"trade-off"* será diferente para cada situação, entretanto, para tomar essa decisão é necessário entender o comportamento e os efeitos que o BC tem. Para tal fim, criaram-se diferentes cenários de programação do BC. Estes são analisados e comparados durante a etapa de execução.

Inicialmente se trabalha com o programa de base que considera BC de 25%, a Figura 4.20 mostra o programa correspondente à etapa final da execução. Repare-se como a inserção desse tamanho de BC provoca um grau de isolamento de projetos superior à necessidade do sistema, fazendo com que o consumo dos BP dos projetos dois e três seja mínimo. O tempo não utilizado do BP é análogo à matéria prima desperdiçada num sistema de manufatura, o tempo e recursos disponibilizados no projeto sobre-dimensionado poderiam ter sido alocados em algum outro projeto.

A partir disso surge o questionamento de como saber oportunamente se os projetos precisam de mais ou menos segurança em relação à interação com outros projetos do sistema. Mediante o uso de "programas híbridos",que em vez de inserir os BC no programa de base os insere no programa projetado, permite-se observar, no primeiro dia execução, o efeito que o deslocamento dos projetos (ocasionado pelos BC) teria sobre os BP

Essa análise demonstra alguns indícios que ajudam a determinar o tamanho do BC. Ao observar o efeito que este tem no consumo dos BP, pode-se ter uma idéia da dimensão do BC como complemento de segurança do BP na proteção do projeto. Quer dizer que, se as inserções dos BC no programa projetado provocam o consumo total (ou grande parte) do BP, é muito provável que o BC esteja sobre-

dimensionado. Da mesma maneira, se o consumo do BP é nulo (ou pouco) é possível que o projeto precise de maior proteção para agüentar os efeitos das interdependências.

Essa lógica pode ser observada no programa híbrido da Figura 4.21, onde o BP do projeto dois é ultrapassado pela programação projetada da terceira atividade, enquanto o consumo do BP do projeto 3 oscila em torno de 50%. Tais características indicam que o espaço gerado pelo BC que protege o projeto dois está claramente sobre-dimensionado. Isto se explica observando o recurso gargalo nos projetos um e dois. Repare-se que, mesmo compartilhando o recurso "A" não precisamente interdependem por causa dele. Isso porque a disponibilidade do gargalo suporta as atividades de ambos os recursos, fazendo com que a necessidade do BC seja unicamente para nivelar os recursos não gargalos. No entanto, o projeto três interdepende de todos os projetos que o precedem, motivo pelo qual, precisaria um BC maior do que o projeto dois. Lembre-se que o BC dimensionado a 25% não deu indícios de estar errando nem por falta nem por excesso, isto é corroborado na Figura 4.20, onde o BP chega a ter certo grau de funcionalidade.

Essa programação híbrida sugere que a determinação dos BC teria que ser dependente das características dos projetos. Contudo, a ferramenta de *software* obriga que os BC sejam dimensionados unicamente com base no recurso gargalo, sem permitir utilizar diferentes percentagens de dimensionamento para cada projeto. Entretanto, a metodologia não se aprofunda nos critérios e métodos de dimensionamento dos BC. A análise da relação entre os BC e as características dos projetos é aprofundada no Capítulo V.

ID	Task Name	Focus Dur	% Impact	Reduction Impact	Days Slack	Check Task	Chain Left	Orig. Chain
1	**Multiproject_Pristsker**	**11 days**	**0**	**0**	**0**			
1	Cero	0 days	0	0	0			
2	**Atividade um**	**4 days**	**0**	**100**	**0**			
3	**Atividade dois**	**3 days**	**0**	**100**	**0**			
4	Atividade tres	3 days	-100	0	4			
5	FB|Atividade tres-3|Fim-5	2 days	-100	0	0	4-3;Atividade tres	3d	3d
6	**Fim**	**0 days**	**0**	**100**	**0**			
7	PB|Fim-5	4 days	0	0	0	2-1;Atividade um	7d	7d
2	**Multiproject_Pristsker 2**	**8 days**	**0**	**0**	**0**			
1	**Cero**	**0 days**	**0**	**50**	**0**			
2	**Atividade 1**	**3 days**	**0**	**50**	**0**			
3	Atividade 2	2 days	-33	50	1			
4	FB|Atividade 2-3|Fim-5	1 day	0	0	0	3-3;Atividade 2	2d	2d
5	**Atividade 3**	**2 days**	**0**	**50**	**0**			
6	**Fim**	**0 days**	**0**	**100**	**0**			
7	PB|Fim-5	3 days	0	0	0	1-1;Cero	5d	5d
3	**Multiproject_Pristsker 3**	**8 days?**	**0**	**0**	**0**			
1	**Cero**	**0 days**	**-67**	**0**	**2**			
2	**Atividade 1**	**5 days**	**0**	**100**	**0**			
3	Atividade 2	1 day?	-133	0	4			
4	FB|Atividade 2-3|Fim-4	1 day	-300	0	0	3-3;Atividade 2	1d	1d
5	**Fim**	**0 days**	**0**	**100**	**0**			
6	PB|Fim-4	3 days	0	0	0	2-2;Atividade 1	5d	5d

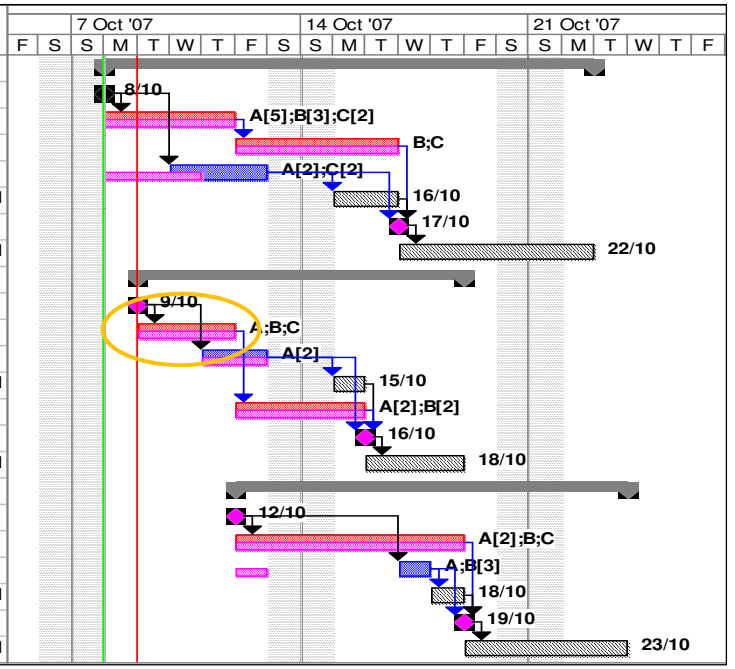

Figura 4.13: Programa master baseado no recurso gargalo "A". Status do dia 8/10

A data de início do projeto dois é antecipada dois dias em relação á figura 4.11. Repare-se que o *programa projetado* permite sobrecargas dos recursos não gargalos (C e B). Isso define uma data de término mais cedo para o projeto 2 (18/10).

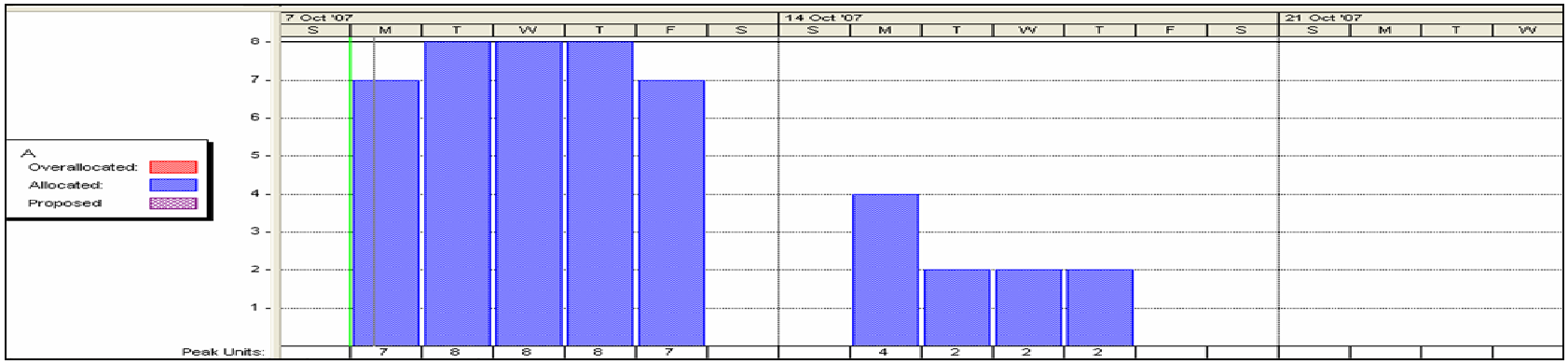

Figura 4.14: Diagrama de carga de trabalho, Recurso "A" (Programa sem BC)

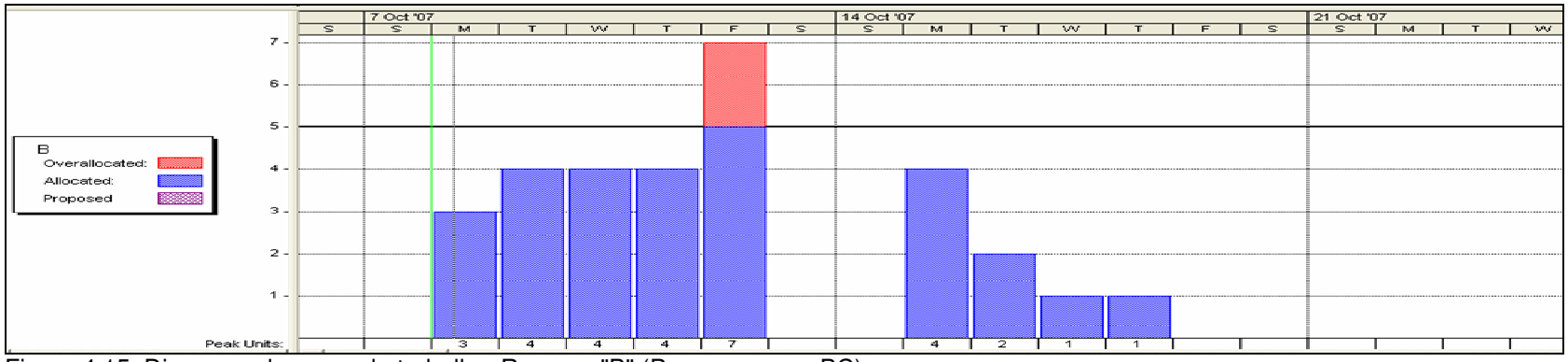

Figura 4.15: Diagrama de carga de trabalho, Recurso "B" (Programa sem BC)

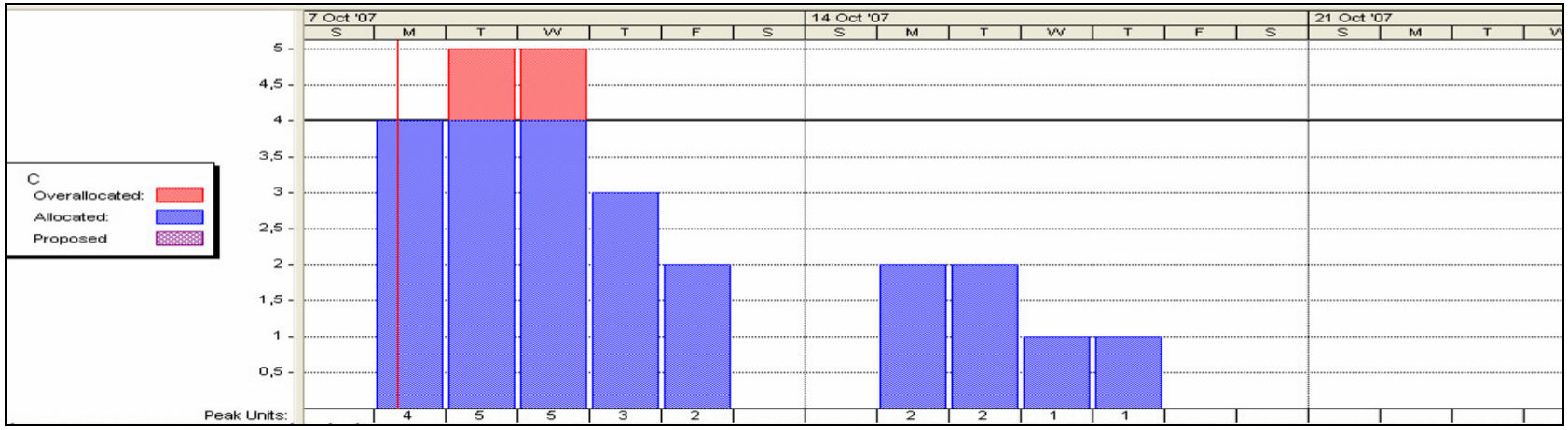

Figura 4.16: Diagrama de carga de trabalho, Recurso "C"(Programa sem BC)

ID	Task Name	Focus Dur	% Impact	Reduction Impact	Days Slack	Check Task	Chain Left	Orig. Chain
1	Multiproject_Pristsker	11 days	0	0	0			
1	Cero	0 days	0	0	0			
2	Atividade um	4 days	0	100	0			
3	Atividade dois	3 days	0	100	0			
4	Atividade tres	3 days	-100	0	4			
5	FB|Atividade tres-3|Fim-	2 days	-100	0	0	4-3;Atividade tres	3d	3d
6	Fim	0 days	0	100	0			
7	PB|Fim-5	4 days	0	0	0	2-1;Atividade um	7d	7d
2	Multiproject_Pristsker 2	8 days	0	0	0			
1	Cero	0 days	-111	0	3			
2	Atividade 1	3 days	0	100	0			
3	Atividade 2	2 days	-100	0	3			
4	FB|Atividade 2-3|Fim-5	1 day	-200	0	0	3-3;Atividade 2	2d	2d
5	Atividade 3	2 days	0	100	0			
6	Fim	0 days	0	100	0			
7	PB|Fim-5	3 days	0	0	0	2-2;Atividade 1	5d	5d
3	Multiproject_Pristsker 3	8 days?	0	0	0			
1	Cero	0 days	-102	0	3			
2	Atividade 1	5 days	0	100	0			
3	Atividade 2	1 day?	-133	0	4			
4	FB|Atividade 2-3|Fim -4	1 day	-300	0	0	3-3;Atividade 2	1d	1d
5	Fim	0 days	0	100	0			
6	PB|Fim -4	3 days	0	0	0	2-2;Atividade 1	5d	5d

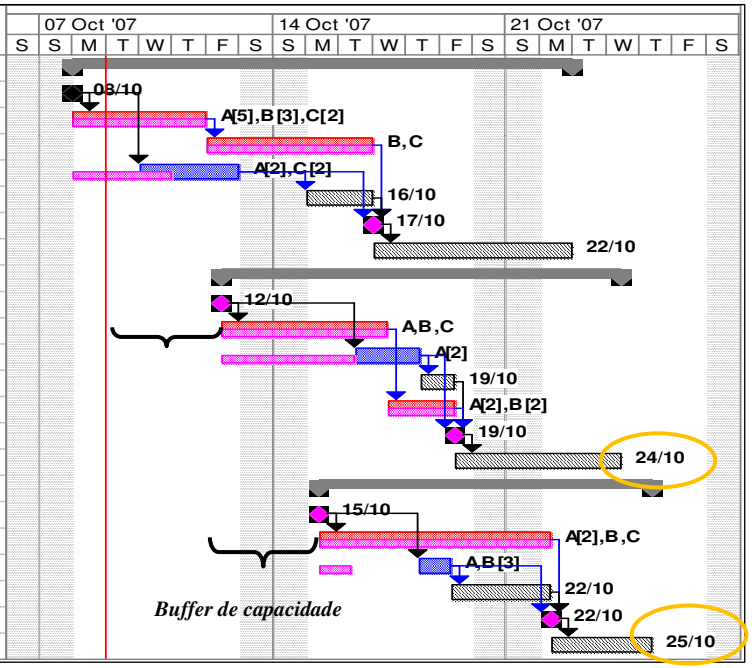

Figura 4.17: Programa máster com BC ao 25% . Status do dia 8/10

A inserção de BC definidos em 25% da carga do recurso gargalo (A), faz que o projeto 2 seja deslocado do 9/10 ao 12/10, e que o projeto 3 seja deslocado do 12/10 ao 15/10. Isto ocasiona que o *makespan* do sistema seja incrementado em 14 dias, (do 8/10 ao 25/10), fazendo que as datas de término dos projetos 2 e 3 sejam mais tarde.

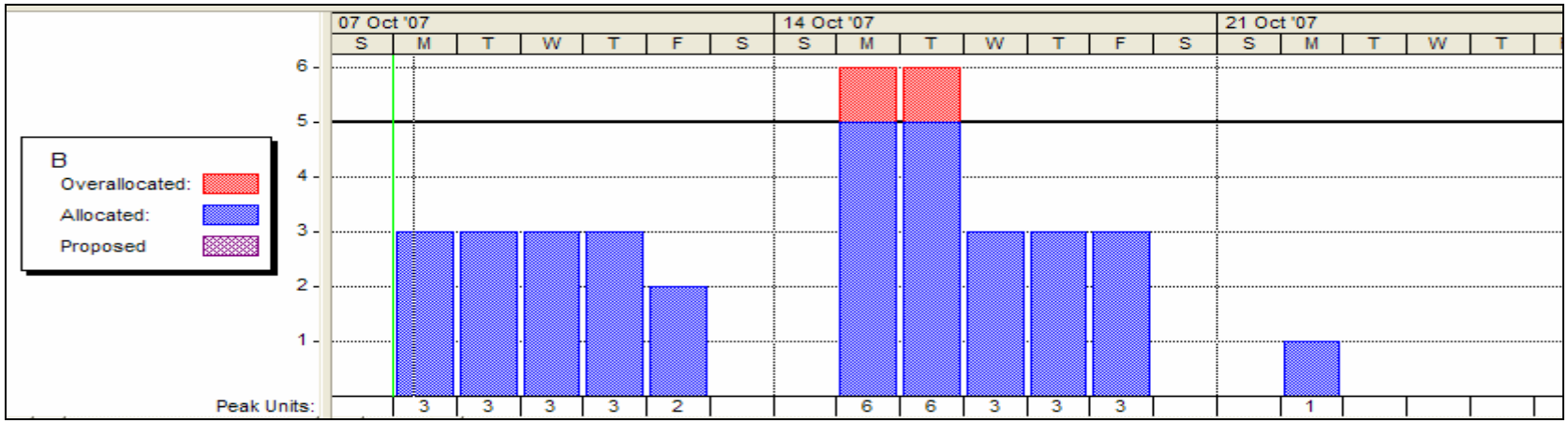

Figura 4.18: Diagrama de carga de trabalho, Recurso "B" (Programa com BC 25%)

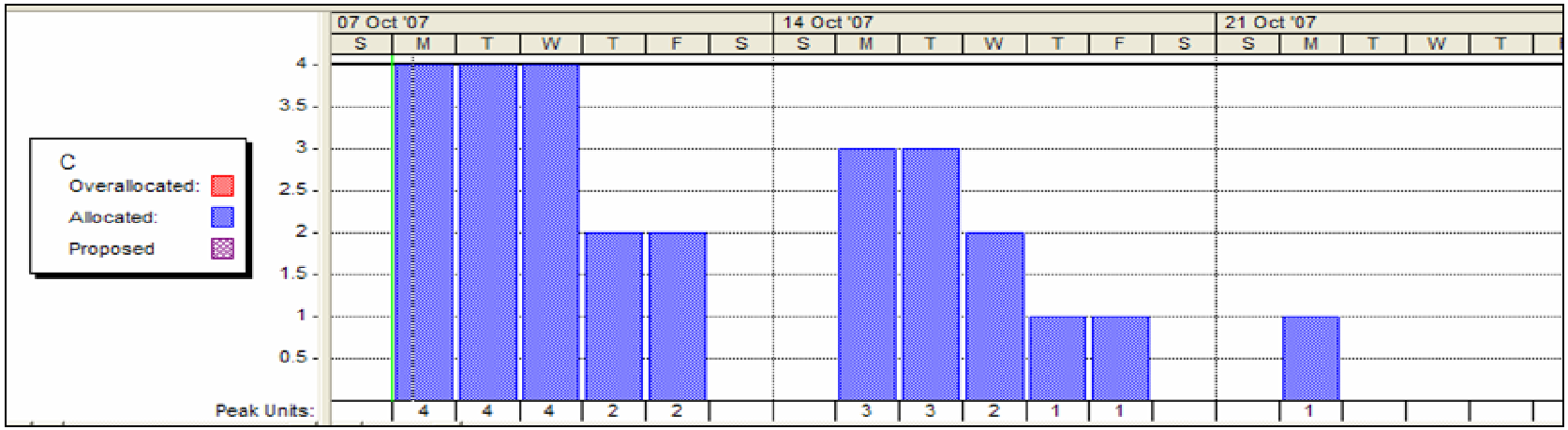

Figura 4.19: Diagrama de carga de trabalho, Recurso "C" (Programa com BC 25%)

ID	Task Name	Focus Dur	% Impac	Reduction Impact	Days Slack	Check Task	Chain Left	Orig. Chair
1	Multiproject_Pristsker	11 days	0	0	0			
1	Cero	0 days	0	0	0			
2	Atividade um	5 days	0	100	0			
3	Atividade dois	3 days	25	100	0			
4	Atividade tres	4 days	0	0	0			
5	FB|Atividade tres-3|Fim-	2 days	-50	0	0		0d	3d
6	Fim	0 days	25	100	0			
7	PB|Fim-5	4 days	25	0	0	3-2;Atividade dois	0.99d	7d
2	Multiproject_Pristsker 2	8.33 days	0	0	0			
1	Cero	0 days	0	0	0			
2	Atividade 1	3 days	0	100	0			
3	Atividade 2	2 days	0	0	0			
4	FB|Atividade 2-3|Fim-5	1 day	-233	0	0		0d	2d
5	Atividade 3	2 days	-11	100	0			
6	Fim	0 days	-11	100	0			
7	PB|Fim-5	3 days	-11	0	0	5-4;Atividade 3	2d	5d
3	Multiproject_Pristsker 3	8.06 days?	0	0	0			
1	Cero	0 days	0	0	0			
2	Atividade 1	5 days	39	100	0			
3	Atividade 2	1 day?	-61	0	3			
4	FB|Atividade 2-3|Fim -4	1 day	-84	0	0	3-3;Atividade 2	1d	1d
5	Fim	0 days	39	100	0			
6	PB|Fim -4	3 days	39	0	0	2-2;Atividade 1	4d	5d

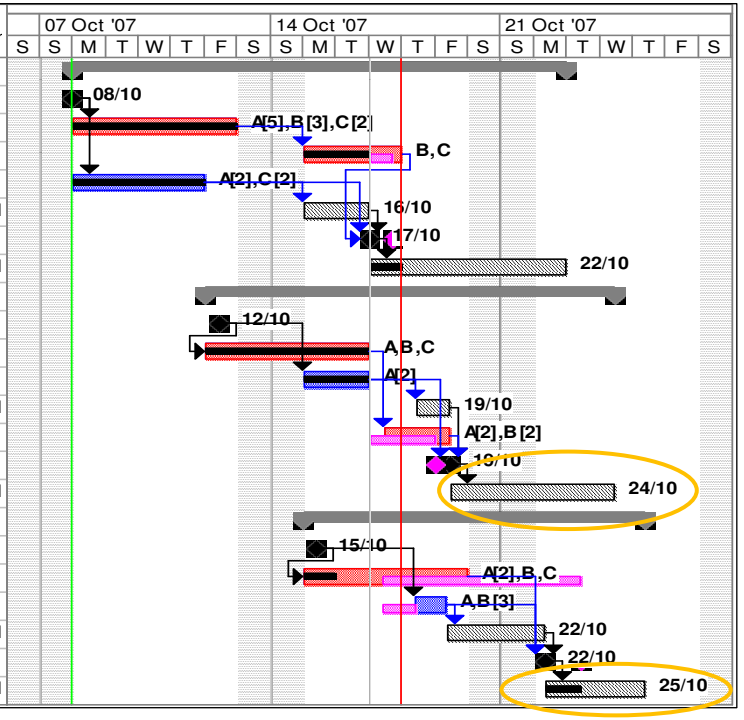

Figura 4.20: Programa máster com BC 25%. Status do dia 17/10

A inserção de BC definidos em 25% isola os projetos, evitando que estes tenham alto grau de compartilhamento de recursos. Não obstante, atrasando as mesmas atividades do que nos casos anteriores se percebe que na etapa final de execução o BP do projeto dois se mantém inativo. Ou seja, houve uma alocação de tempo no projeto que foi desperdiçada, podendo ter se alocado nesse tempo outros projetos ou atividades. O BP do projeto 3 apresenta um consumo provocado unicamente pelas interdependências indicando que o complemento de segurança que o BC lhe da é necessário.

ID	Task Name	Focus Dur	% Impact	Reductio Impact	Days Slack	Check Task	Chain Left	Orig. Chain
1	**Multiproject_Pristsker**	**11 days**	**0**	**0**	**0**			
1	Cero	0 days	0	0	0			
2	**Atividade um**	**4 days**	**0**	**100**	**0**			
3	**Atividade dois**	**3 days**	**0**	**100**	**0**			
4	Atividade tres	3 days	0	0	0			
5	FB|Atividade tres-3|Fim-5	2 days	-50	0	0	4-3;Atividade tres	3d	3d
6	**Fim**	**0 days**	**0**	**100**	**0**			
7	PB|Fim-5	4 days	0	0	0	2-1,4-3;Atividade um	6d	7d
2	**Multiproject_Pristsker 2**	**8 days**	**0**	**0**	**0**			
1	**Cero**	**0 days**	**0**	**50**	**0**			
2	**Atividade 1**	**3 days**	**100**	**50**	**0**			
3	Atividade 2	2 days	17	50	3			
4	FB|Atividade 2-3|Fim-5	1 day	150	0	0	3-3;Atividade 2	2d	2d
5	**Atividade 3**	**2 days**	**100**	**50**	**0**			
6	**Fim**	**0 days**	**100**	**100**	**0**			
7	PB|Fim-5	3 days	117	0	0	2-2;Atividade 1	5d	5d
3	**Multiproject_Pristsker 3**	**8 days?**	**0**	**0**	**0**			
1	**Cero**	**0 days**	**0**	**0**	**0**			
2	**Atividade 1**	**5 days**	**41**	**100**	**0**			
3	Atividade 2	1 day?	-92	0	4			
4	FB|Atividade 2-3|Fim-4	1 day	-177	0	0	3-3;Atividade 2	1d	1d
5	**Fim**	**0 days**	**41**	**100**	**0**			
6	PB|Fim-4	3 days	41	0	0	2-2;Atividade 1	5d	5d

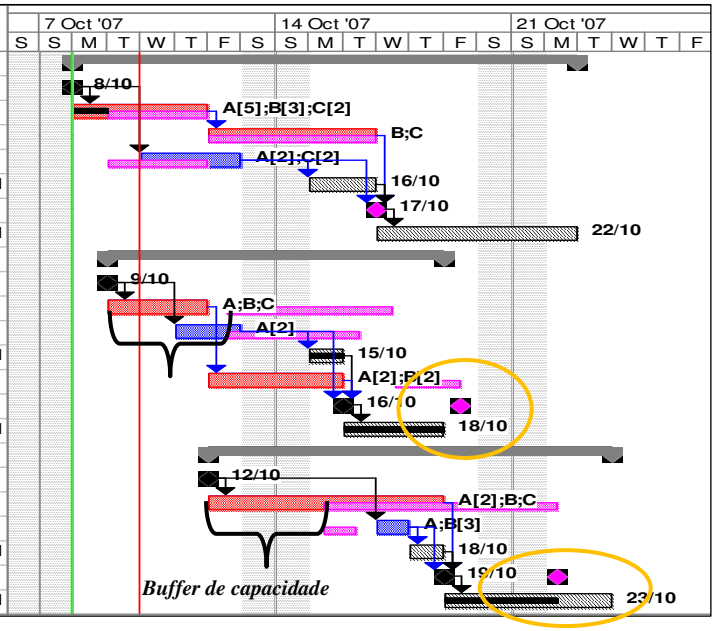

Figura 4.21: Programa máster híbrido. Status do dia 09/10 (Programa projetado com BC 25%)

O programa híbrido em vez de inserir o BC dentro do *programa de linha de base* o insere no *programa projetado*. Repare-se como existe um espaçamento entre o início das atividades de linha de base e as projetadas. Isso permite observar, no primeiro dia de execução, a relação de tamanho que existe entre os BC e BP em função do consumo que o primeiro ocasiona no segundo. Veja como no projeto 2 o consumo do BP ultrapassa o programado, indicando que o BC está sobre-dimensionado. Ao fazer esta comparação se espera que o consumo dos BP ocasionado pelos BC não seja próximo do total nem do zero.
A necessidade que cada projeto tem do BC é em diferentes dimensões.